

### Adv DevOps Exp 07

**Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.**

#### Step 1:

Ensure Installations and path settings of the following softwares/services:

1. **Jenkins**
2. **Docker Desktop Engine**

Both these services help us create a sonarqube image and build a project within sonarqube using the pipeline architecture in jenkins.



**Run this command in powershell administrator mode:** `docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest`

This kickstarts the Sonarqube service/server at post **9000**. It also creates a Docker image within our Docker Desktop account. Make sure to keep your Docker Engine up and running

Powershell output:

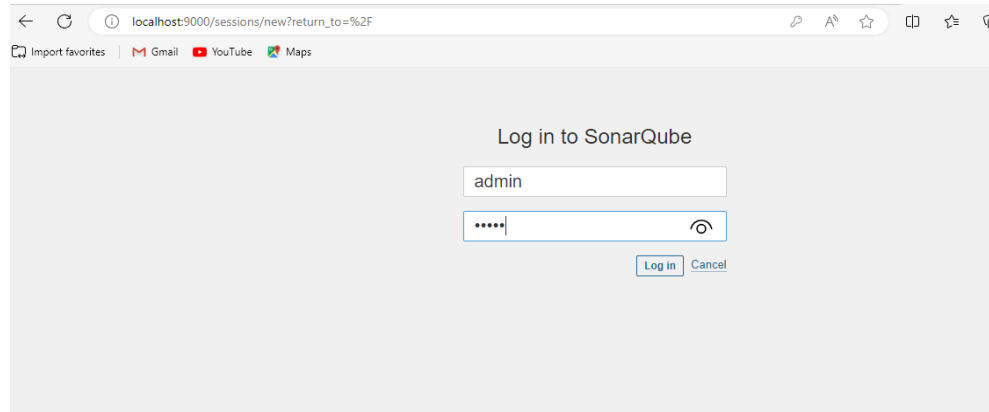
```
PS C:\Users\INF505-16> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
2d9047cbfbab55bd03ed72f7349feebf5d79434b23ff7eda329694bd24115349
PS C:\Users\INF505-16> |
```

Sonarqube started on Docker:

<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started
<input type="checkbox"/>	 <a href="#">sonarqube</a> 2d9047cbfbab	<a href="#">sonarqube:latest</a>	Running	282.07%	<a href="#">9000:9000</a> 	12 seconds ago

Go to localhost:9000. On fulfilling the desired action, we arrive on a page that asks us for our username and password.

The default credentials for Sonarqube server are (username,password) == **(admin,admin)**



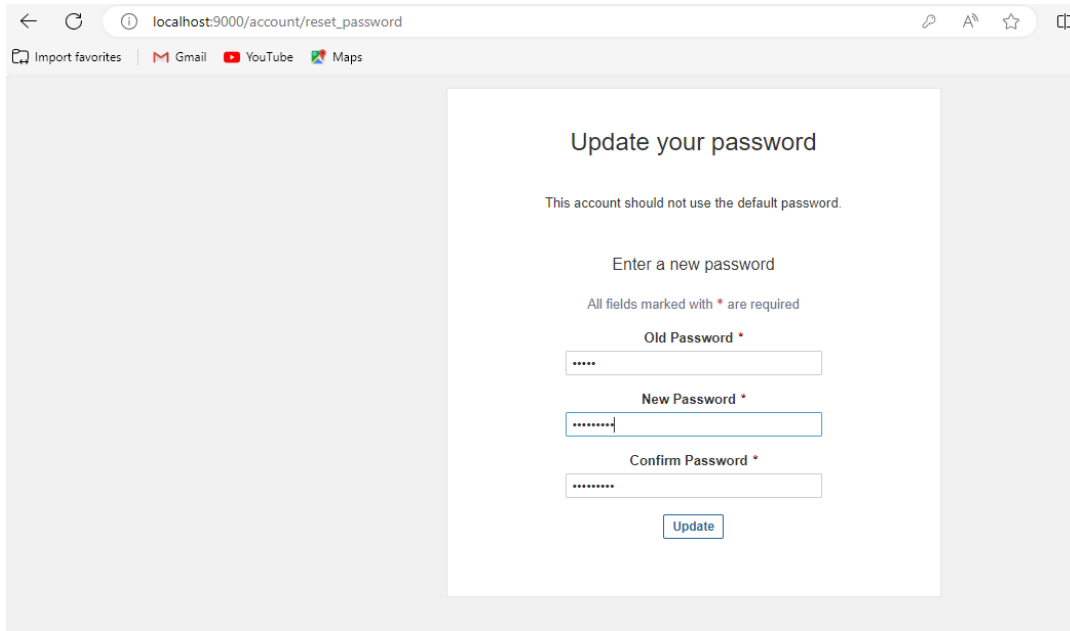
Log in to SonarQube

admin

.....

Log in Cancel

Sonarqube asks us to update the default password and make a new password for this Sonarqube account. So i entered a new suitable password (keep it easy enough for one to remember)



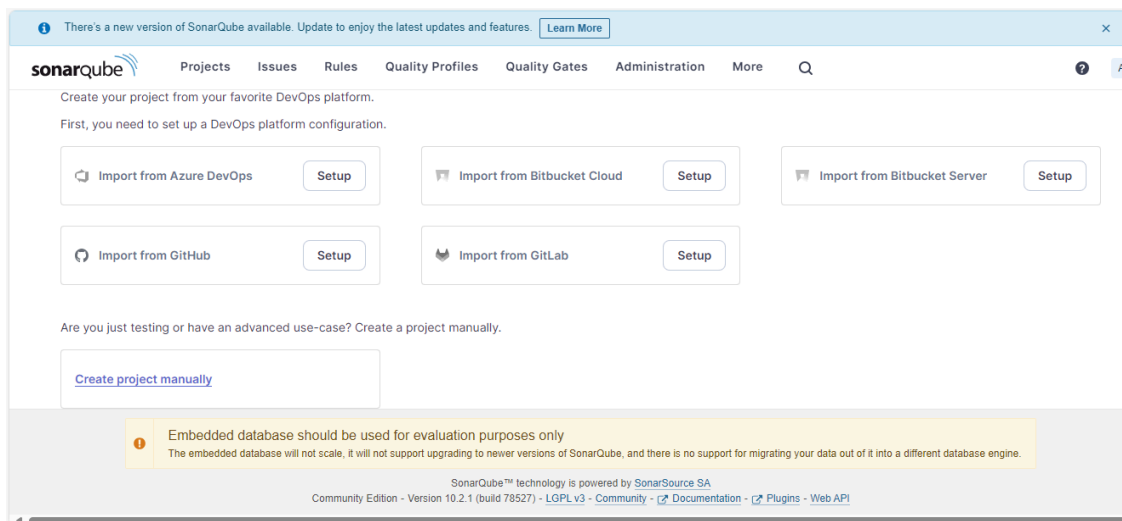
The screenshot shows a web browser window with the address bar displaying 'localhost:9000/account/reset\_password'. The page title is 'Update your password'. Below the title, a message states: 'This account should not use the default password.' The main heading is 'Enter a new password'. A note below says 'All fields marked with \* are required'. There are three input fields: 'Old Password \*' (containing '\*\*\*\*\*'), 'New Password \*' (containing '\*\*\*\*\*'), and 'Confirm Password \*' (containing '\*\*\*\*\*'). An 'Update' button is located at the bottom of the form.

## Step 2:

After logging in,

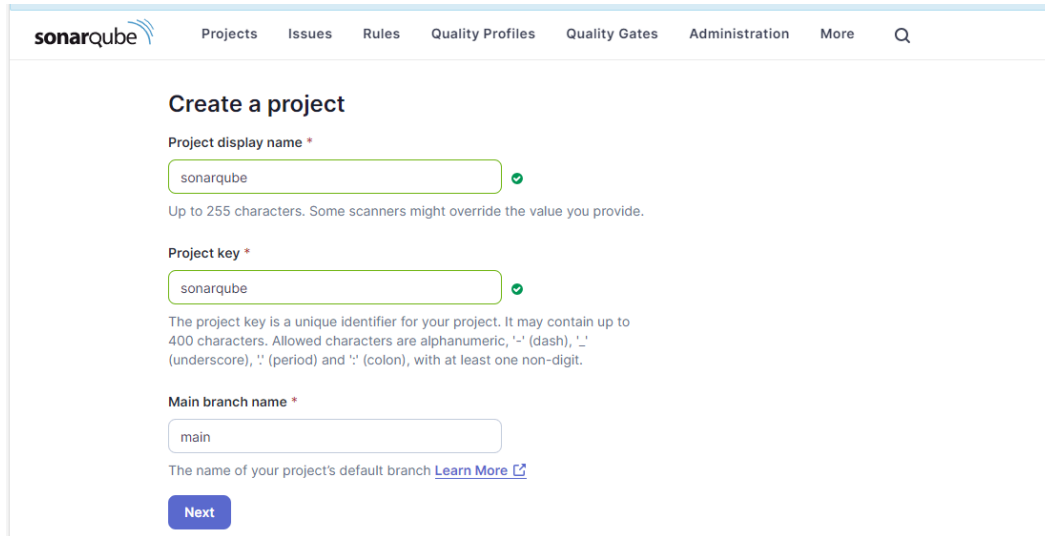
This is the page that we land on... This tells us that with the help of docker we have successfully created a Sonarqube image and that we have assigned a port number 9000 to it so as to use its services and build a Sonarqube project.

From here on, we are supposed to make a new project manually by selecting the 'Create project manually' option



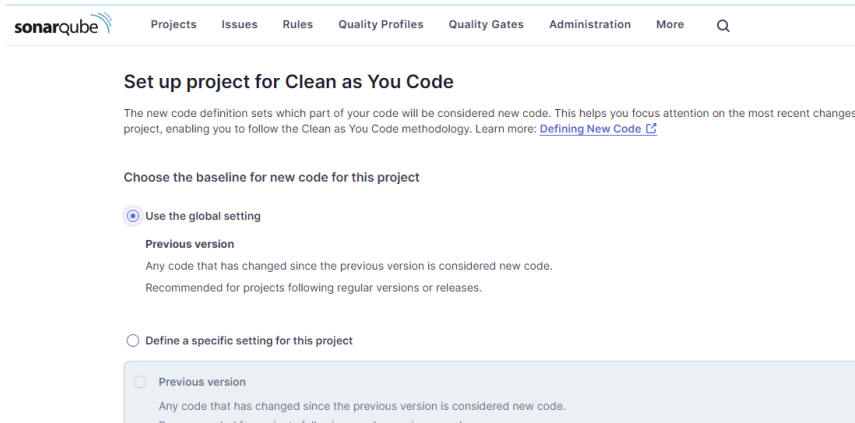
The screenshot shows the Sonarqube dashboard. At the top, there is a notification bar: 'There's a new version of SonarQube available. Update to enjoy the latest updates and features. Learn More'. Below this is the navigation bar with links: Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search icon. The main content area has the heading 'Create your project from your favorite DevOps platform.' and a sub-heading 'First, you need to set up a DevOps platform configuration.' There are five buttons for importing projects: 'Import from Azure DevOps', 'Import from Bitbucket Cloud', 'Import from Bitbucket Server', 'Import from GitHub', and 'Import from GitLab'. Each button has a 'Setup' button next to it. Below these buttons, there is a link 'Create project manually'. At the bottom, there is a yellow warning box: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' The footer contains the text: 'SonarQube™ technology is powered by SonarSource SA. Community Edition - Version 10.2.1 (build 78527) - LGPL v3 - Community - Documentation - Plugins - Web API'.

We are now supposed to enter the name for our Sonarqube  
Lets keep it as 'sonarqube'



The screenshot shows the 'Create a project' page in SonarQube. It has a navigation bar with links: Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search icon. The main heading is 'Create a project'. There are three input fields: 'Project display name' with the value 'sonarqube' and a green checkmark, 'Project key' with the value 'sonarqube' and a green checkmark, and 'Main branch name' with the value 'main'. Below the first two fields is a note: 'Up to 255 characters. Some scanners might override the value you provide.' Below the third field is a note: 'The name of your project's default branch [Learn More](#)'. At the bottom is a blue 'Next' button.

Next, we are asked for setting up our sonarqube project  
We will select global settings for this project and finally create the project named 'sonarqube'



The screenshot shows the 'Set up project for Clean as You Code' page in SonarQube. It has the same navigation bar as the previous page. The main heading is 'Set up project for Clean as You Code'. Below it is a paragraph: 'The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes in your project, enabling you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)'. Below this is a section 'Choose the baseline for new code for this project'. There are two radio buttons: 'Use the global setting' (selected) and 'Define a specific setting for this project'. Under 'Use the global setting' are two options: 'Previous version' (selected) and 'Any code that has changed since the previous version is considered new code. Recommended for projects following regular versions or releases.' Under 'Define a specific setting for this project' is one option: 'Previous version' (unselected) and 'Any code that has changed since the previous version is considered new code. Recommended for projects following regular versions or releases.'

**Step 3:** Go to Jenkins Dashboard and download the plugin named '**Sonarqube Scanner for Jenkins**'. This plugin will help us to build our Sonarqube project through a project created within Jenkins. Make sure to restart Jenkins once installation is complete and Enable this plugin to use its services



The screenshot shows the 'Plugins' page in Jenkins. On the left is a sidebar with links: Updates (with a red badge '57'), Available plugins, Installed plugins (selected), and Advanced settings. On the right is a search bar with the text 'sonar'. Below the search bar is a table with columns 'Name' and 'Enabled'. The table has one row: 'SonarQube Scanner for Jenkins 2.17.2'. The description for this plugin is: 'This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin'. The 'Enabled' column has a toggle switch that is turned on (blue) and a red 'X' icon.

Go to Jenkins system configurations, and navigate to Sonarqube installation  
Enter the Sonarqube's project name, the URL and server authentication token if needed

Dashboard > Manage Jenkins > System >

### SonarQube installations

List of SonarQube installations

**Name**

**Server URL**  
Default is http://localhost:9000

**Server authentication token**  
SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

+ Add

Save

Apply

Select 'install automatically' option for installation of the specified Sonarqube Scanner version

☒ Install automatically ?

**Install from Maven Central**

**Version**

SonarQube Scanner 6.1.0.4477

Add Installer

Add SonarQube Scanner

Save

Apply

**Step 4:** Make a freestyle project in Jenkins and give it a suitable name. This project is being made by us, so as to carry out our build of our Sonarqube project and perform static analysis of it based on whether the build is successful or not.


**Jenkins** Search (CTRL+K)


Dashboard > All > New Item


### New Item

Enter an item name

Select an item type

**Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments.

OK

Choose this GitHub repository in Source Code Management.

[https://github.com/shazforiot/MSBuild\\_firstproject.git](https://github.com/shazforiot/MSBuild_firstproject.git)

It is a sample hello-world project with no vulnerabilities and issues, just to test

Git ?

Repositories ?

Repository URL ?

[https://github.com/shazforiot/MSBuild\\_firstproject](https://github.com/shazforiot/MSBuild_firstproject)

Credentials ?

- none -

+ Add

Advanced

Add Repository

Save Apply

Next, under build steps, select 'Execute Sonarqube Scanner' option

Dashboard > project1 > Configuration

## Configure

- General
- Source Code Management
- Build Triggers
- Build Environment**
- Build Steps
- Post-build Actions

☐ Inspect build log for published build scans

☐ Terminate a build if it's stuck

☐ With Ant ?

### Build Steps

Add build step ^

Filter

- Execute SonarQube Scanner
- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit
- SonarScanner for MSBuild - Begin Analysis

There is an option for entering 'Analysis properties' in which i entered the following credentials/parameters related to my Sonarqube account or the project that i have made:

```
sonar.projectKey=com.example.myproject
sonar.projectName=My Sample Project
sonar.projectVersion=1.0
sonar.sources=.
sonar.host.url=http://localhost:9000
sonar.login=sqa_180e0f1309adfbbaa226d862355a9feb1066eddfd
sonar.projectBaseDir=C:/ProgramData/Jenkins/.jenkins/workspace/project1
```

Dashboard > project1 > Configuration

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

#### Analysis properties ?

```
sonar.projectKey=sonarqube
sonar.projectName=sonarqube
sonar.projectVersion=1.0
sonar.sources=.
sonar.host.url=http://localhost:9000
sonar.login=sqa_180e0f1309adfbbaa226d862355a9feb1066eddfd
sonar.projectBaseDir=C:/ProgramData/Jenkins/.jenkins/workspace/project1
```

#### Additional arguments ?

#### JVM Options ?

**Step 5:** In the analysis properties, in order to set the **sonar.login** parameter, we are supposed to navigate to our account details and choose the option of generating a global token. Create a new token...

Now, this assists us to identify our project uniquely without having to provide our credentials or in case our credentials fail to be identified

This token creation step is optional and is only to be used when our analysis properties consisting of our normal credentials fail

### Tokens

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

#### Generate Tokens

Name	Type	Expires in	
<input type="text" value="Enter Token Name"/>	<input type="text" value="Select Token Type"/>	<input type="text" value="30 days"/>	<button>Generate</button>

Name	Type	Project	Last use	Created	Expiration	Act
token1	Global		< 1 hour ago	September 23, 2024	October 23, 2024	<button>Revoke</button>

Enter a new password

Now, navigate to security within administration and provide administrator system and execute analysis rights to the entity named admin.

The screenshot shows the SonarQube Administration interface, specifically the Security section. It displays a table of users and their permissions. The 'Administrator' user is highlighted with a green background.

User	Administer System	Administer	Execute Analysis	Create
<b>sonar-administrators</b> System administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input checked="" type="checkbox"/> Projects
<b>sonar-users</b> Every authenticated user automatically belongs to this group	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
<b>Anyone</b> <span>DEPRECATED</span> Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects
<b>A Administrator</b> admin	<input checked="" type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects

4 of 4 shown

After Building...the console output should indicate success

This indicates that the image that we created in docker successfully led us to creating a project within sonarqube, which in turn led us to building it over jenkins by uniquely identifying our sonarqube project. Build stage of our sonarqube project in jenkins involves static analysis for which we passed in the appropriate properties

The screenshot shows the Jenkins build interface for build #6. The build is successful, indicated by a green checkmark. The build details include the revision, repository, and build data.

**Jenkins** Search (CTRL+K) ?

Dashboard > project1 > #6

**Status** ☒ #6 (Sep 23, 2024, 2:24:02 PM) [Add description](#) [Keep this build forever](#)

**Changes** **Console Output** **Edit Build Information** **Delete build '#6'** **Git Build Data** **Previous Build** **Next Build**

**Started by anonymous user** Started 11 min ago Took 13 sec

**git** **Revision:** f2bc042c04c6e72427c380bcae6d6fee7b49adf **Repository:** [https://github.com/shazforiot/MSBuild\\_firstproject](https://github.com/shazforiot/MSBuild_firstproject)

- refs/remotes/origin/master

**No changes.**

**Conclusion:** The integration of Jenkins with SonarQube for static analysis using Docker provides a streamlined and automated approach to ensuring code quality. By successfully configuring the necessary tools—Docker for hosting SonarQube and Jenkins for building and analyzing the project—you can perform seamless static analysis using SonarQube's powerful scanning capabilities. This setup automated vulnerability detection, ensuring that potential issues are caught early. The use of GitHub as a repository for code allows for real-time scanning and continuous integration, ensuring that projects meet quality standards before moving forward in the development lifecycle.